11-2017

Predicting indoor crowd density using column-structured deep neural network

Akihito SUDO

Teck Hou (DENG Dehao) TENG
Singapore Management University, thteng@smu.edu.sg

Hoong Chuin LAU
Singapore Management University, hclau@smu.edu.sg

Yoshihide SEKIMOTO

DOI: https://doi.org/10.1145/3152341.3152349

Published in PredictGIS 2017: Proceedings of the 1st ACM SIGSPATIAL Workshop on Prediction of Human Mobility, Redondo Beach, CA, November 7-10. https://doi.org/10.1145/3152341.3152349

Predicting Indoor Crowd Density using Column-Structured Deep Neural Network

Akihito Sudo
Shizuoko University
Shizuoka-Shi, Shizuoka-Ken, Japan
sudo@inf.shizuoka.ac.jp

Teck-Hou Teng
Singapore Management University
Singapore
dengdehao@gmail.com

Hoong Chuin Lau
Singapore Management University
Singapore
hclau@smu.edu.sg

Yoshihide Sekimoto
The University of Tokyo
Tokyo, Japan
sekimoto@iis.u-tokyo.ac.jp

ABSTRACT

This work proposes a deep neural network approach known as the column-structured deep neural network (COL-DNN-$\mathcal{R}$) for predicting crowd density in an indoor environment using historical Wi-Fi traces of individual visitors. With a structure designed to minimize feature engineering, COL-DNN accepts raw features such as crowd density, opening and closing hours and peak visitor counts for extracting features. The extracted features are used by a regression model $\mathcal{R}$ for predicting the crowd densities. Standard regression models such as MLP, RF and SVM can be used as $\mathcal{R}$. Experiments are performed to investigate the effect of feature representation and model structure on the prediction accuracy. Experiment results show the best prediction accuracy is obtained using features extracted by COL-DNN and using MLP as the regression model, i.e., $\mathcal{R} = MLP$.

CCS CONCEPTS

• **Information Systems** → **Geographic Information Systems**; *Information Systems Applications*; *Spatial-Temporal Systems*;

KEYWORDS

Indoor Crowd Prediction, Deep Neural Network, Feature Extraction

ACM Reference Format:
Akihito Sudo, Teck-Hou Teng, Hoong Chuin Lau, and Yoshihide Sekimoto. 2017. Predicting Indoor Crowd Density using Column-Structured Deep Neural Network. In *Proceedings of 1st ACM SIGSPATIAL Workshop on Prediction of Human Mobility (PredictGIS 2017),* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Crowding is a side effect experienced by popular venues. It decreases economic value and experience of visitors. Taken to the extreme, it may lead to severe incidents such as a stampede. To improve the safety and the experience of their visitors, venue operators strive to pre-empt or respond effectively to large crowd as early as possible. Hence, accurate look-ahead prediction of crowd densities plays a crucial role in this aspect of their daily operational needs.

Naive formulation of crowd density prediction entails a regression task predicting future crowd densities using past crowd densities. Due to the challenge of building a good feature vector for each venue manually, many recent works [15] do not attempt to develop their crowd estimation models using additional features about the crowd. Our approach overcomes this challenge by building feature vectors for any venue automatically. And unlike computer vision-based solutions like [5, 14, 16], our approach uses Wi-Fi traces of visitor positions over time. Thus, our approach is not constrained by physical layouts and quality of videos.

Motivated by the recent applications of deep neural networks (DNN) [3, 11], we propose a column-structured DNN (COL-DNN-$\mathcal{R}$) for predicting indoor crowd densities. Considered a first use of DNN for crowd density prediction, COL-DNN is structured using *prior knowledge* of input features common to the venues. The column structure simplifies the network structure and reduces the number of parameters to optimize. Consequentially, such a network structure simplifies search for the optimal parameters, reduces the training duration and achieves good prediction accuracy while minimizing feature engineering.

To evaluate and compare the performance of our proposed approach, experiments were conducted using real Wi-Fi dataset comprising records of visitors' positions obtained from a large venue operator. The records were aggregated to give the hourly crowd densities at the venues. The trained model predicted crowd density at selected venues from one to five hours into the future. Compared with the benchmark models, our experiment results show that COL-DNN-*MLP* has the best prediction accuracy.

The contributions of this paper are as follows:

- Our proposed approach derives additional features on the crowd automatically.
- In the experiments using real Wi-Fi datasets, we improve the prediction accuracy over existing approaches by 20.07% using the additional features.
- We further improve the prediction accuracy by 24.35% using COL-DNN-*MLP*.

2 RELATED WORK

A recent survey [15] of approaches for estimating crowd density and size of crowd reveals several works based on visual inspection of crowded scenes. Examples of such works include [6] and [16]. [6] uses convolutional neural network (CNN) to estimate crowd density. Some network connections are removed and two CNNs are cascaded to improve classification accuracy and speed. [16] uses CNNs for crowd counting. CNN is pre-trained using whole images to derive high level features. The features are mapped using recurrent network layers with memory cells to local counting numbers. The output quality from such approaches is typically affected by the quality of the crowd imagery. Our work side-steps such challenges by using Wi-Fi data comprising position records of the visitors.

Other approaches on estimating crowd densities are also known. For instance, [5] generates crowd density maps automatically using local features. The generated crowd density maps are used to enhance detection and tracking of high density crowds and study the effect of different obfuscation levels on the context-awareness of the crowd. Local crowd density is used with regular motion patterns for crowd change detection and event recognition. [14] evaluates several regression models and feature types for determining the size of crowd. That work finds local features that yield better results than the holistic features or histogram features. [7] explores the use of pseudo-landmark points to improve the prediction accuracy without increasing the cost of prediction. Unlike these works, the structure of our proposed COL-DNN-*R* is suitable for extracting non-visual data automatically.

Systems built for studying crowd movement are also known. For instance, [2] proposes an approach for gathering large dataset using mobile application. The collected data is analyzed and used for planning program and perimeter design. [1] presents a hybrid indoor/outdoor positioning and navigation system. User mobility patterns are mapped for predicting potential bottlenecks and hotspots. [10] introduces Indoor-ALPS. This location prediction system uses temporal-spatial features on an ensemble of four classifiers to create individual daily models that predict movement of users. Unlike these works, we minimize the effort spent on feature engineering by proposing DNN with structure suitable for non-image data.

[13] propose a multi-factor neural network attention model for fusing multiple groups of features by training it with a hidden representation-based attention mechanism. It is shown to perform better than the HME gating network [8]. Like [13], our neural network (NN)-based approach uses features represented using vectors but minimizes feature engineering of the features. No need for an attention mechanism seen in [13], our NN-based approach is simpler and more straight-forward. The weights and parameters are tuned by error-backpropagation and a gradient descent method. We find our NN-based approach to be capable of producing the desired responses after training.

3 PROBLEM STATEMENT

This work addresses the problem of predicting future values of aperiodic time series. The shape of the density chart depends on time and space. Each data point may differ from the seasonal data. Hence, this is a prediction problem of irregular seasonal time series.

Formally, the future crowd densities $\mathbf{y}_{r,t} = \{y_{r,t+1}, \cdots, y_{r,t+h_A}\}$ is represented as a probability distribution $P(\mathbf{y}_{r,t}|\mathbf{y}^{prev}_{r,t}, \{\mathbf{w}^j_{r,t}\}_{j\in[1:P]})$ where r denotes the venue, t denotes the time, $\mathbf{y}^{prev}_{r,t} = \{y_{r,t}, \cdots, y_{r,t-h_B}\}$ denotes the previous crowd densities, $y_{r,t-\tau}$ denotes the crowd density τ time steps before t, $\mathbf{w}^j_{r,t}$ denotes the set of P additional features, h_A denotes the look-ahead hours and h_B denotes the number of preceding hours.

For instance, given $P = 3$ additional features of opening hour, closing hour, and peak count of crowd, one representation of $\{\mathbf{w}^j_{r,t}\}_{i=\{1,2,3\}}$ is $\mathbf{w}^j_{r,t} = \{0, \cdots, 0, 1, 0, \cdots, 0\} \in \mathbb{B}^{24}$ for $i = \{1, 2\}$ and the number of peak count $\mathbf{w}^3_{r,t} \in \mathbb{N}$.

The previous crowd densities $\mathbf{y}^{prev}_{r,t}$ and the set of additional features $\mathbf{w}^j_{r,t}$ can be represented as a set of vectors $\{\mathbf{x}^i\}_{i\in[1:P+1]}$, where $\mathbf{x}^1_{r,t} = \mathbf{y}^{prev}_{r,t}$ and $\mathbf{x}^i_{r,t} = \mathbf{w}^j_{r,t}$ for $j \in [1 : P]$. Then, the problem can be viewed as building a regression model which predicts $\mathbf{y}_{r,t}$ using $\{\mathbf{x}^i_{r,t}\}_{i\in[1:P+1]}$ for all r.

4 PREDICTION OF CROWD DENSITY

To predict crowd density with minimum feature engineering, we propose a modular deep neural network (DNN) architecture known as COL-DNN-$\mathcal{R}$. From Figure 1, the modules of COL-DNN-$\mathcal{R}$ are the Feature Extraction Module (FEM) and the Regression Module (RM). The weights of neurons in FEM are trained offline using previous crowd densities $\mathbf{y}^{prev}_{r,t}$ and the set of additional features $\mathbf{w}^i_{r,t}$. Training of regression model $\mathcal{R}$ in RM commences only after the weights of neurons in the FEM are trained.

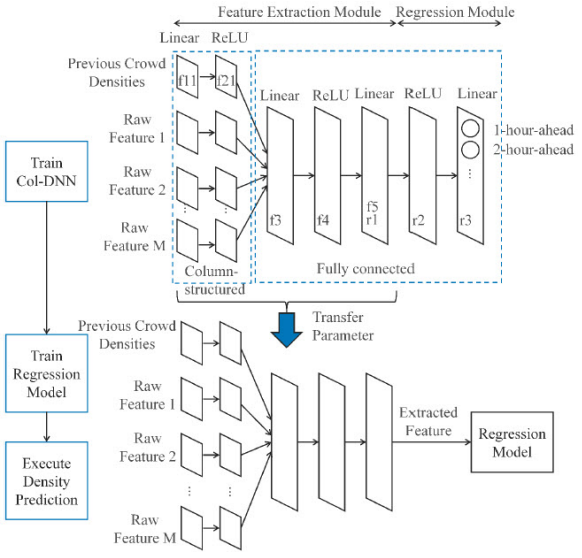

Figure 1: Modular architecture of COL-DNN-$\mathcal{R}$ comprising FEM and RM.

4.1 Feature Extraction Module

As seen in Figure 1, the Feature Extraction Module (FEM) has two column-structured layers (f1,f2) and three fully connected layers

(f3,f4,f5). Raw features entering the column-structured layers are transformed and channeled to the fully-connected layers. It enters the RM comprising two fully-connected layers and a regression model. We call features coming from the FEM as the extracted features $\mathbf{x}^{ext}$. The structure of FEM is based on the types and the dimension of data. This work uses raw features comprising the previous crowd densities, the opening hour and the closing hour and peak crowd count to predict the crowd densities.

Neurons in the f1 column-structured layer (input layer) use linear activation function while neurons in the f2 column-structured layer (hidden layer) use Rectified linear unit (ReLU) as the activation function. Neurons at the fully-connected f3,f4, f5 layers are interleaved between the linear and the ReLU activation functions. Such a design of a five-layer FEM has lesser number of weights compared to a five-layer fully-connected neural network. Consequentially, time to train the FEM to reach convergence is reduced.

The number of columns is M and the number of the input neurons at the i-th column is N_i^x when data is $\{\mathbf{x}^i | i \in [1:M]\}$ ($\mathbf{x}^i \in \mathbb{R}^{N_i^x}$). Each column-structured layer is matched to a numerical input vector. The number of types of inputs determines the number of columns, i.e., M.

FEM propagates the raw features to the RM in the following manner. Given $\{\mathbf{x}^i\}_{i\in[1:M]}$, the outputs of the f2 column-structured layer is

$$\mathbf{y}^{f2,i} = f^{ReLU}(W^{f1,i}\mathbf{x}^i + \mathbf{b}^{f1,i}), \tag{1}$$

where $f^{ReLU}(x) = max(0, x)$, $W^{f1,i} \in \mathbb{R}^{N_i^x} \times \mathbb{R}^{N^{f2,i}}$ is the weights between f1 and f2 column-structured layer and $\mathbf{b}^{f1,i} \in \mathbb{R}$ is the bias. There are $N^{f2,i}$ hidden neurons in the f2 column-structured layer. The input of the f3 fully-connected layer is obtained by combining $\mathbf{y}^{f2,i}$ using

$$\mathbf{x}^{f3} = \sum_{i=1}^{M}(W^{f2,i}\mathbf{y}^{f2,i} + \mathbf{b}^{f2,i}). \tag{2}$$

The activation functions of the f3 layer is the identity function. The same structure and process as a standard multi-layer perceptron (MLP) is employed for f3, f4, and f5 layers. Hence, the output of FEM y^{f5} is

$$\mathbf{y}^{f5} = MLP^f(\mathbf{x}^{f3}), \tag{3}$$

where MLP^f is the function mapping the input to the output of a 3-layer MLP.

$$MLP^f(\mathbf{x}) = W^{f4} f^{ReLU}(W^{f3}\mathbf{x} + \mathbf{b}^{f3}) + \mathbf{b}^{f4}. \tag{4}$$

The output of FEM y^{f5} is also referred to as the extracted feature $\mathbf{x}^{ext}$ to the RM.

4.2 Regression Module

The Regression Module (RM) has the regression model $\mathcal{R}$ which is trained for prediction using $\mathbf{x}^{ext}$. The layers have the same structure and process as a standard MLP. Yet, it can predict more accurately than just MLP without the FEM. This is possible because it accepts $\mathbf{x}^{ext}$ as the input.

ReLU is used as the activation function of the hidden neurons in the r2 layer. At the r3 layer, the number of neurons is dependent on the number of outputs. For instance, the output of m^{th} neuron at the final layer is the predicted crowd density m hours later. Hence, the predicted crowd densities from $1 - h_A$ hours later is presented as follows:

$$\mathbf{y}^{pred} = (y_1^{pred}, y_2^{pred}, \cdots, y_{h_A}^{pred}) = MLP^r(\mathbf{x}^{ext}) \tag{5}$$

where y_m^{pred} is the predicted density m hours later, and MLP^r represents the map of MLP as follows:

$$MLP^r(\mathbf{x}) = W^{r2} f^{ReLU}(W^{r2}\mathbf{x} + \mathbf{b}^{r1}) + \mathbf{b}^{r2} \tag{6}$$

Knowing that fully connected layers r1, r2 and r3 is actually an MLP, it can be regarded as a regression model $\mathcal{R}$. This means $\mathcal{R}$ can be any other regression models such as support vector machines (SVM) and random forests (RF). If another regression model is used with the trained FEM, the new regression model has to be trained before it can be used for prediction. As seen in bottom of Figure 1, the input to RM is the extracted features $\mathbf{x}^{ext}$.

4.3 Training

Our proposed COL-DNN-$\mathcal{R}$ is trained in two phases using the same training data. The first training phase trains the weights of the neurons in f1-f5 of FEM and r1-r3 of RM using $\mathcal{R} \equiv MLP$. If $\mathcal{R} \neq MLP$, the new $\mathcal{R}$ is trained in a second training phase using the extracted features $\mathbf{x}^{ext}$ as the inputs.

Algorithm 1 Algorithm for training COL-DNN-$\mathcal{R}$ with $\mathcal{R} = MLP$.

Require: Training data $\mathcal{X}^{train} \times \mathcal{Y}^{train}$
1: Initialize $\mathcal{W}^f = \{\{W^{f1,i}\}_{i\in[1:M]}, W^{f2}, W^{f3}, W^{f4}\}$
2: Initialize $\mathcal{W}^r(\mathcal{R}) = \{W^{r1}, W^{r2}\}$
3: **repeat**
4: **for** $(\{\mathbf{x}^i\}_{i\in[1:M]}, \mathbf{y})$ **in** $\mathcal{X}^{train} \times \mathcal{Y}^{train}$ **do**
5: Tune $\mathcal{W}^f$ and $\mathcal{W}^r(\mathcal{R})$ to fit $(\{\mathbf{x}^i\}_{i\in[1:M]}, \mathbf{y})$ by gradient descent
6: **end for**
7: **until** Convergence of $\mathcal{W}^f, \mathcal{W}^r(\mathcal{R})$
8: **return** Tuned $\mathcal{W}^f, \mathcal{W}^r(\mathcal{R})$

The *first training phase* where COL-DNN-$\mathcal{R}$ is trained using supervised learning is outlined using Algorithm 1. COL-DNN-$\mathcal{R}$ for $\mathcal{R} = MLP$ is trained to fit $\mathcal{Y}^{train}$. Being a directed acyclic graph and having differentiable activation functions, the gradient of the weights are derived by the back-propagation method. Then, the weights can be optimized by a gradient descent method such as stochastic gradient descent, Adagrad or Adam [4, 9].

A *second training phase* follows when $\mathcal{R} \neq MLP$. The tuning of the parameters of $\mathcal{R}$ using the same training data $\mathcal{X}^{train} \times \mathcal{Y}^{train}$ is outlined using Algorithm 2. In this training phase, the weights $\mathcal{W}^f$ of the neurons in the FEM are fixed. The raw input $\mathcal{X}^{train}$ is presented to the FEM to give $\mathbf{x}^{ext}$. To fit $\mathcal{Y}^{train}$, the parameters $\mathcal{W}^r(\mathcal{R})$ of regression model $\mathcal{R}$ are tuned using $\mathbf{x}^{ext}$.

4.4 Prediction

COL-DNN-$\mathcal{R}$ is trained for predicting $\mathbf{y}^{pred}$ following the process outlined using Algorithm 3. The prediction output $\mathbf{y}^{pred}$ is semantically identical to $\mathcal{Y}^{train}$ seen in Algorithm 1 and Algorithm 2.

Algorithm 2 Algorithm for training $\mathcal{R}$ when $\mathcal{R} \neq MLP$.

Require: Training data $\mathcal{X}^{train} \times \mathcal{Y}^{train}$
Require: Tuned $\mathcal{W}^f$
1: Initialize regression model $\mathcal{R}$
2: **repeat**
3: **for** $(\{\mathbf{x}^i\}_{i\in[1:M]}, \mathbf{y})$ **in** $\mathcal{X}^{train} \times \mathcal{Y}^{train}$ **do**
4: Calculate $\mathbf{x}^{ext}$ using $\mathbf{x}^i$ and $\mathcal{W}^f$ in Eq. 3
5: Tune $\mathcal{W}^r(\mathcal{R})$ to fit $(\mathbf{x}^{ext}, \mathbf{y})$
6: **end for**
7: **until** Convergence of $\mathcal{W}^r(\mathcal{R})$
8: **return** Tuned $\mathcal{W}^r(\mathcal{R})$

Algorithm 3 Algorithm for predicting $\mathbf{y}^{pred}$ using trained COL-DNN-$\mathcal{R}$.

Require: Test data $\{\mathbf{x}^{i,test}\}_{i\in[1:M]}$
Require: Tuned weights $\mathcal{W}^f$ of neurons in FEM
Require: Trained regression model $\mathcal{R}$
1: Derive $\mathbf{x}^{ext}$ using $\mathbf{x}^{i,test}$ and $\mathcal{W}^f$ in Eq. 3
2: Present $\mathbf{x}^{ext}$ to $\mathcal{R}$ for predicting $\mathbf{y}^{pred}$
3: **return** Prediction output $\mathbf{y}^{pred}$

Keeping $\mathcal{W}^f$ and $\mathcal{W}^r(\mathcal{R})$ stationary, the trained COL-DNN-$\mathcal{R}$ predicts $\mathbf{y}^{pred}$ using $\{\mathbf{x}^{i,test}\}_{i\in[1:M]}$ as the input. From Algorithm 3, the prediction process involves forward-passing $\mathbf{x}^{i,test}$ through FEM to give $\mathbf{x}^{ext}$. RM accepts $\mathbf{x}^{ext}$ as the inputs for predicting $\mathbf{y}^{pred}$.

5 DESIGN OF EXPERIMENTS

The design choices of the experiments are presented here.

5.1 Wi-Fi Dataset

The raw Wi-Fi datasets comprises the position records $x_{r,t}$ of people at venue r at time t. This information is collected at $5-6$ minutes interval. The positions are approximated based on the strength of the Wi-Fi signal omitted from the mobile devices of the people. Each position record has data fields like time, user ID and location ID. Time is a string with the format YYYY-MM-DD HH:MM:SS, user ID is an alpha-numeric string of 64 characters and location ID is a numeric array of 10 characters.

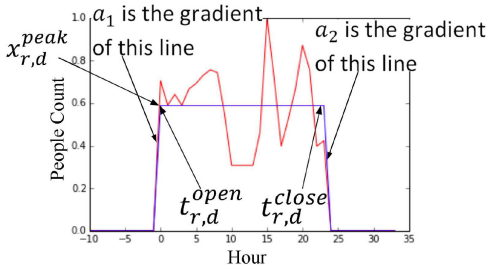

Figure 2: Derivation of additional features such as opening hour $T^{open}_{r,d}$, closing hour $T^{close}_{r,d}$ and peak counts $X^{peak}_{r,d}$.

In addition to the Wi-Fi dataset, the prediction models uses additional features such as the opening hour $T^{open}_{r,d}$, the closing hour $T^{close}_{r,d}$ and the peak counts $X^{peak}_{r,d}$. From Figure 2, these features are derived using a fitting Gaussian $N(\mu_{r,d}(t;\theta_{r,d,a_i},\sigma_{r,d}))$ to the density data $X_{r,d,t}$ where $\mu_{r,d,t}(t;\theta_{r,d},a_i)$ is defined as $\mu_{r,d,t}(t;\theta_{r,d},a_i) = X^{peak}_{r,d} * ReLu(-ReLu(-a_1(t - T^{open}_{r,d})) + 1) * ReLu(-ReLu(a_2(t - T^{close}_{r,d})) + 1)$ and $ReLu(t) = \max(0, t)$.

5.2 Aggregation of Data Records

The observed crowd densities $\mathbf{y}^{prev}_{r,t}$ are derived from the Wi-Fi dataset. To do that, the raw position records have to be interpolated to the hourly marks first because those position records are not necessarily available at the desired time marks. The raw position records are interpolated by assuming the position of a person seen at the time closest to the desired time mark remains unchanged.

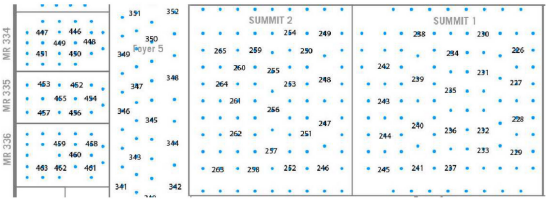

Figure 3: Illustration of a small part of a particular level of building covered by the Wi-Fi dataset.

After that, the interpolated position records are aggregated to give the observed crowd count $c_{r,t,p}$ at position p in venue r at time t. Figure 3 illustrates a number of positions in various venues on a particular level of a building. The observed crowd densities $\mathbf{y}^{prev}_{r,t}$ are derived using $\frac{\sum_p^{P_r}\{c_{r,t,p}\}}{a_r}$ where a_r is the area and P_r is the number of positions in venue r.

5.3 Crowd Density

The predicted crowd densities $\mathbf{y}^{pred}_{r,t}$ is a vector of scalar values representing the crowd densities at fixed time intervals. The predicted crowd densities $\mathbf{y}^{pred}_{r,t}$ is validated against the observed crowd densities $\mathbf{y}_{r,t}$ like those seen in Figure 4 where the crowd densities in selected large and small venues are illustrated. It shows the crowd density at one hour interval. Larger fluctuation of crowd density is seen for the larger venues.

Prediction accuracy of the trained prediction models with respect to x_i is shown as the root mean square error (RMSE) derived using $\sqrt{\frac{\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}{n}}$ where n is the number of predicted values. For predicting crowd density at 1hr to 5hr look-ahead choices, we have $n = 5$.

5.4 Parameter Settings

Parameter settings used for the prediction models are presented here.
COL-DNN-$\mathcal{R}$: There are 72 neurons for f21, 24 neurons for f22 and f23, 1 neuron for f24, 121 neurons for f3, f4, f5 (r1) and r2. The gradient descent method used is the Adam [9] algorithm with the

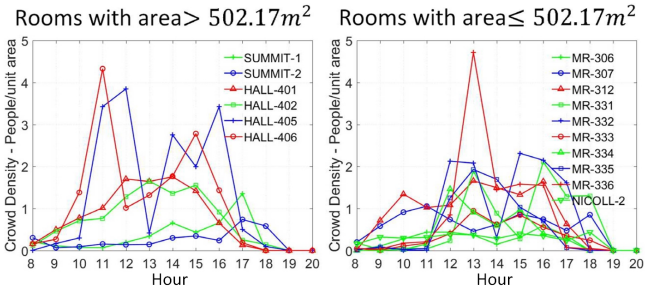

Figure 4: Illustration of crowd densities in large and small venues from 0800hr to 2100hr on 18^{th} December 2015. The average venue size is $502.17m^2$

following parameter values: $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e-08$. The size of each batch is 100. COL-DNN-$\mathcal{R}$ is trained for 500 epoches.

Random Forests (RF): This RF combines the classifiers by averaging their probabilistic prediction. The hyper-parameters of RF are tuned using the grid search. The ranges of parameters are $[10, 100, 1,000, 10,000]$ for the number of estimator. The max features are $[1, 21, 41]$ for when only the F1 feature is used and $[1, 21, 41, 61, 81, 101, 121]$ for when the F1-F4 features are used.

Support Vector Regressor (SVR): This SVR has a Radial Basis Function (RBF) kernel. The parameters C, γ and ϵ are tuned using the grid search. The ranges of the parameters are $[10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1]$ for γ, $[1, 10, 100, 1,000, 10,000]$ for C and $[10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1]$ for ϵ.

Multi-layer Perceptron (MLP): This MLP uses an implementation of back-propagation algorithm based on the Adam algorithm [9]. The Adam algorithm has the following parameter values: $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e-08$. The size of each batch is 100. The MLP is trained for 500 epochs.

Seasonal ARIMA (SARIMA): This SARIMA fits an $ARIMA(p, d, q)$ model by an exact maximum likelihood via the Kalman filter. It is tuned using the Hyndman-Khandakar algorithm. The parameter set giving the maximum Akaike's information criterion is used in the experiments.

5.5 Feature Representation

The feature representations seen in Table 1 are used here. F1 denotes the previous people count. It has a dimension of 72 elements because F1 is hourly people count three days, i.e., 72 hr, before the time of prediction. It is standardized using $x_i^{'} = \frac{x_i - \bar{x}_i}{\sigma}$. F2 denotes the opening hour while F3 denotes the closing hour. F2 and F3 are binary vectors of length 24. Each element of F2 and F3 represents an one-hour time slot. The opening and closing hour are indicated using $'1'$ in F2 and F3 respectively. The other times are represented using $'0'$. F4 is an integer number denoting the peak count of people at the venue on that day.

6 PERFORMANCE EVALUATION

Experiments were conducted to evaluate and compare the performance of COL-DNN. The experiment results presented here are the

Table 1: Feature representations considered in this work. *Extracted* features refers to features extracted by COL-DNN using F1-F4 as raw input features

Model	$F1$	$F2$	$F3$	$F4$	Type
SARIMA	•				Raw
F1-MLP	•				Raw
F1-F4-MLP	•	•	•	•	Raw
COL-DNN-MLP	•	•	•	•	Extracted
F1-RF	•				Raw
F1-F4-RF	•	•	•	•	Raw
COL-DNN-RF	•	•	•	•	Extracted
F1-SVR	•				Raw
F1-F4-SVR	•	•	•	•	Raw
COL-DNN-SVR	•	•	•	•	Extracted

RMSEs of the prediction models. Regression models (MLP, SVR and RF) from [12] are used while SARIMA from [17] is used.

Eight months of Wi-Fi datasets (Apr'15 - Nov'15) are used to train the prediction models. The trained prediction models are tested using Wi-Fi dataset of December 2015. The Wi-Fi datasets are cleaned by removing the anomalous records. For illustration purpose, only crowd densities in eight venues of an indoor environment are predicted.

Table 2: Mean RMSE of 10 prediction models for 1-5 hours look-ahead prediction. Ext. denotes the extracted features.

Feature	Model	1-hr	2-hr	3-hr	4-hr	5-hr	mean
Ext.	MLP	**5.88**	**5.96**	**5.90**	**5.78**	**5.65**	**5.83**
Ext.	RF	6.19	6.33	6.29	6.18	6.07	6.21
Ext.	SVR	6.03	6.10	6.05	5.95	5.83	6.00
F1-F4	MLP	6.66	6.64	6.63	6.45	6.34	6.54
F1-F4	RF	6.27	6.41	6.46	6.44	6.39	6.39
F1-F4	SVR	6.11	6.39	6.48	6.52	6.51	6.40
F1	MLP	9.13	8.98	8.83	8.69	8.56	8.84
F1	RF	7.10	7.74	8.01	8.10	8.10	7.81
F1	SVR	7.19	7.87	8.23	8.39	8.44	8.02
F1	SARIMA	6.76	9.60	11.65	13.65	14.92	11.31

The mean RMSEs of the 10 prediction models seen in Table 2 are derived using the RMSEs of the predicted crowd densities from 0800 hr to 2100 hr of day 1 to day 29 of December 2015. The prediction accuracies for the F1-based prediction models deteriorates as the number of look-ahead hours increases. The performance of the F1-F4-based prediction models is more stable than the F1-based prediction models because the weights are tuned using more features. The prediction accuracies improve further for the Ext-based prediction models because COL-DNN reduces the search space for the best features.

Focusing on the effect of feature representations, the plots in Figure 5 and the left plot of Figure 6 directly compare the RMSEs of the same model structure using different feature representations. The plots show the same model structure can have different RMSEs when paired with different feature representations. It is also observed that the RMSEs for the prediction models at the earlier hours are higher because the arrival rate of the visitors are higher at these

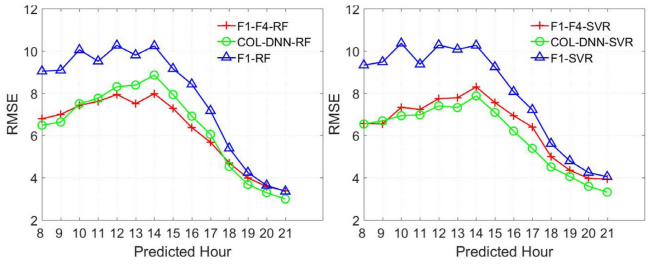

Figure 5: The RMSEs of same model structure with different feature representations.

hours. The RMSEs of COL-DNN-MLP and COL-DNN-SVR are improved more because the extracted features have similar structure to MLP and SVR.

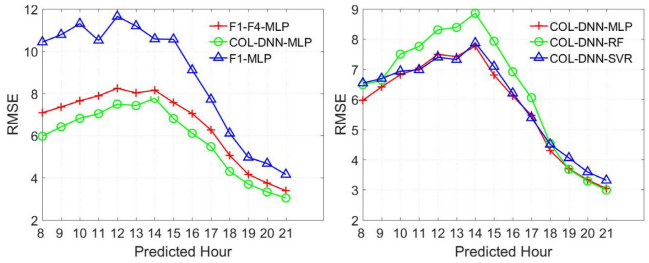

Figure 6: Left: Effects of feature representation on MLP-based model structures; Right: Effects of extracted feature on different model structures.

Focusing on the effect of model structures, the plots in Figure 7 and the right plot of Figure 6 directly compare the RMSEs of different model structures paired with the same feature representation. It can be observed from these plots that the model structures have rather distinct response to the same feature representation. Such observation implies the feature representations have larger influence than the model structures on the prediction accuracies. In addition, the F1-based SARIMA and COL-DNN-RF are observed performing poorly. Only from 1800 hr onward, COL-DNN-RF is performing as well as COL-DNN-MLP. Similar levels of performance are observed for the F1-F4-based prediction models. This is because the structure of RF is fundamentally different from that of MLP and SVR.

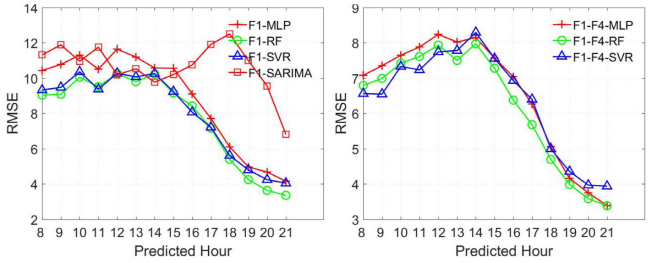

Figure 7: The RMSEs of different model structures with same feature representation.

Figure 8 compares the aggregated RMSEs of COL-DNN-MLP with F1-F4-MLP, F1-F4-RF and F1-F4-SVR. The left plot shows COL-DNN-MLP is consistently much accurate than the other prediction models for predicting the hourly crowd densities. The right plot shows COL-DNN-MLP has the lowest RMSEs consistently for most days of December. This is because the size of crowd is small enough for the prediction models to perform well in similar ways. More specifically, the separation of the RMSEs among these approaches are the most distinct at 2-3, 7-8 and 18-20 of December. According to event calendar of the venue, these dates turn out to be the days with large events.

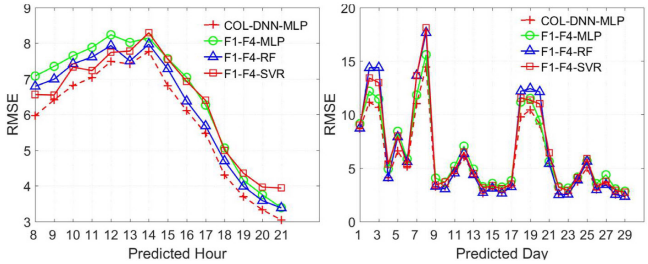

Figure 8: The RMSEs of COL-DNN-MLP and state-of-the-art benchmarks (F1-F4-MLP, F1-F4-RF and F1-F4-SVR) for hourly and daily prediction of crowd densities.

The improvement ratios seen in Table 3 further confirm our COL-DNN-MLP is better at predicting crowd densities at large events. This is a desirable traits because such a solution is more needed at the larger events where the crowd is larger and more acute problems can occur. The observations imply a correct pairing of feature representation (COL-DNN) and model structure (MLP) gives the best prediction accuracy of crowd density.

Table 3: Improving ratio (%) of COL-DNN-MLP over selected benchmark methods

Benchmark	Large Events	Small Events
F1-F4-MLP	7.65	14.83
F1-F4-RF	19.17	2.32
F1-F4-SVR	15.60	10.13

7 SUMMARY AND CONCLUSION

This work addresses a prediction problem on irregular seasonal time series. It is contextualized to the setting of predicting crowd densities of indoor environment over several look-ahead hours at the same time. Side-stepping challenges encountered by the computer vision-based approaches, this work uses Wi-Fi dataset comprising position records of visitors. The proposed methodology is a modular architecture comprising the Feature Extraction Module (FEM) and the Regression Module (RM) known as COL-DNN-$\mathcal{R}$. The FEM has two column-structured and three fully connected layers. It extracts features from the raw input features to be used as inputs to the RM. At the RM, a trained regression model predicts the crowd densities at several look-ahead hours.

Experiments were conducted to evaluate and compare performance of COL-DNN-$\mathcal{R}$. Features comprising the previous crowd

densities (PV), the opening hour (OH), the closing hour (CH) and the peak count (PC) of an event are used. PV, OH and CH are used by COL-DNN for feature extractions. The extracted features and PC are used by a regression model such as MLP, SVM, RF or SARIMA for predicting the crowd densities. The experiment results show COL-DNN-MLP gives the best prediction accuracy. It is also found that COL-DNN-MLP is better than the compared benchmarks at predicting crowd densities of large events. Such observations imply good feature representation and model structure are necessary for good prediction accuracy.

This work can be extended at several fronts. At the FEM, further analysis can be performed on the extracted features. It is hoped the analysis can explain how the extracted features improve the prediction accuracy of the regression models. At the RM, regression models can be built using a wider variety of machine learning techniques such as auto-encoders, recurrent neural networks and restricted Boltzmann machines. Last but not least, COL-DNN-$\mathcal{R}$ can surely be scaled up to predict crowd densities at multiple levels, more look-ahead choices and indoor environment with configurable spaces.

ACKNOWLEDGMENTS

This research project is funded by National Research Foundation Singapore under its Corp Lab @ University scheme and Fujitsu Limited.

REFERENCES

[1] G. Biczok, S. Diez Martinez, T. Jelle, and J. Krogstie. 2014. Navigating MazeMap: Indoor human mobility, spatio-logical ties and future potential. In *Proceedings of PERCOM Workshops*. 266–271.

[2] Ulf Blanke, Gerhard Troster, Tobias Franke, and Paul Lukowicz. 2014. Capturing crowd dynamics at large scale events using participatory GPS-localization. In *Proceedings of the* 9^{th} *IEEE International Conference on ISSNIP*. IEEE, 1–7.

[3] Quanjun Chen, Xuan Song, Harutoshi Yamada, and Ryosuke Shibasaki. 2016. Learning deep representation from big and heterogeneous data for traffic accident inference. In *Proceedings of the* 30^{th} *AAAI Conference on Artificial Intelligence*. 338–344.

[4] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[5] Hajer Fradi and Jean-Luc Dugelay. 2015. Towards crowd density-aware video surveillance applications. *Information Fusion* 24 (2015), 3–15.

[6] Min Fu, Pei Xu, Xudong Li, Qihe Liu, Mao Ye, and Ce Zhu. 2015. Fast Crowd Density Estimation with Convolutional Neural Networks. *Engineering Applications of Artificial Intelligence* 43 (2015), 81–88.

[7] Cho-Jui Hsieh, Si Si, and Inderjit S Dhillon. 2014. Fast Prediction for Large-Scale Kernel Machines. In *Advances in Neural Information Processing Systems 27*. 3689–3697.

[8] Michael I Jordan and Robert A Jacobs. 1994. Hierarchical mixtures of experts and the EM algorithm. *Neural computation* 6, 2 (1994), 181–214.

[9] Diederik P. Kingma and Jimmy Lei Ba. 2015. ADAM: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representation*. 1–15.

[10] Christian Koehler, Nikola Banovic, Ian Oakley, Jennifer Mankoff, and Anind K. Dey. 2014. Indoor-alps: an adaptive indoor location prediction system. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 171–181.

[11] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location: a recurrent model with spatial and temporal contexts. In *Proceedings of the* 30^{th} *AAAI Conference on Artificial Intelligence*. 194–200.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[13] Matthew Riemer, Aditya Vempaty, Flavio P Calmon, Fenno F Heath III, Richard Hull, and Elham Khabiri. 2016. Correcting Forecasts with Multifactor Neural Attention. In *Proceedings of the 33rd International Conference on Machine Learning*. 3010–3019.

[14] David Ryan, Simon Denman, Sridha Sridharan, and Clinton Fookes. 2015. An evaluation of crowd counting methods, features and regression models. *Computer Vision and Image Understanding* 130 (2015), 1–17.

[15] Sami Abdulla Mohsen Saleh, Shahrel Azmin Suandi, and Haidi Ibrahim. 2015. Recent Survey on Crowd Density Estimation and Counting for Visual Surveillance. *Engineering Applications of Artificial Intelligence* 41 (Feb 2015), 103–114.

[16] Chong Shang, Haizhou Ai, and Bo Bai. 2016. End-to-end crowd counting via joint learning local and global count. In *Proceedings of the IEEE ICIP*. 1215–1219.

[17] StatsModels. 2016. Autoregressive Integrated Moving Average ARIMA(p,d,q) model. http://statsmodels.sourceforge.net. (2016).